CC4302 Sistemas Operativos

Tarea 7 – Semestre Primavera 2023 – Prof.: Luis Mateu

En esta tarea Ud. deberá programar un driver para Linux que implementa una cuenta regresiva (*count down*). El valor inicial de la cuenta regresiva se escribe en */dev/cdown*. Cada vez que se lee */dev/cdown* se entrega el valor actual de la cuenta regresiva y luego se decrementa en 1. El dispositivo */dev/cdown* debe tener número *major* 61.

El siguiente ejemplo usa los comandos estándares de Linux */bin/echo* y *cat* para demostrar el comportamiento esperado de */dev/cdown* . Su driver debe reproducir exactamente el mismo comportamiento. Si hay aspectos que el ejemplo no aclara, decida Ud. mismo tratando de simplificar su tarea. Las filas de la tabla están ordenadas cronológicamente. Lo que escribió el usuario aparece en **negritas**. Observe que el prompt $ indica cuando debe terminar un comando. Si el prompt $ no aparece es porque hay una llamada al sistema pendiente.

Shell 1	Shell 2	Shell 3	Shell 4
$ **cat < /dev/cdown** (1) $			
$ **echo 3 >/dev/cdown** (2) $			
	$ **cat < /dev/cdown** (3) 3 $		
		$ **cat < /dev/cdown** (3) 2 $	
			$ **cat < /dev/cdown** (3) 1 $
	$ **cat < /dev/cdown** (1) $		
		$ **echo 10 >/dev/cdown** (2) $	
$ **cat < /dev/cdown** (3) 10 $			
		$ **echo hola >/dev/cdown** (4) bash: echo: error de escritura: Argumento invalido $	

Notas:

(1) La cuenta regresiva está en 0. Cada vez que se lee con la cuenta regresiva en 0, se debe leer el fin de archivo (0 bytes leídos). El comando *cat* usa *read* para leer de */dev/cdown*. Esto hará que en el núcleo se invoque su implementación de *read* para el dispositivo */dev/cdown*.

(2) Se escribe un número entero con *write* en */dev/cdown*. Esto hará que que se invoque su implementación de *write* en el núcleo para el dispositivo */dev/cdown*. Ud. debe establecer ese número entero como valor de la cuenta regresiva.

(3) El comando *cat* invocará 2 veces *read* para leer */dev/cdown*. La primera vez Ud. debe entregar una línea con el valor actual de la cuenta regresiva, seguida del *newline*. La segunda vez debe entregar fin de archivo (0 bytes leídos). Debe decrementar en 1 la cuenta regresiva.

(4) Se escribe *hola* con *write* en */dev/cdown*. Esto hará que que se invoque su implementación de *write* en el núcleo para el dispositivo */dev/cdown*. Como *hola* no es un valor numérico Ud. debe hacer que *write* retorne un código de error que indique que el argumento es inválido. El mensaje de error aparecerá en el idioma de su instalación de Linux.

Recursos

Baje de U-cursos el archivo *modules2020-2.tgz* y descomprímalo. Contiene enunciados y soluciones de tareas de semestres anteriores con instrucciones para compilarlas y ejecutarlas (ver archivos *README.txt* en cada directorio). Además se adjunta un tutorial sobre programación de módulos y drivers en Linux. Le será de especial utilidad el directorio *Syncread* con el enunciado y la solución de la tarea 3 del semetre otoño de 2013 (que se vio o se verá en clase auxiliar).

Baje además el archivo *t6.zip* y descomprímalo. Programe su solución en el archivo *cdown.c* del directorio *T6*. Ahí encontrará un *Makefile* para compilar su tarea y las instrucciones para crear el dispositivo */dev/cdown*. Resuelva su tarea usando los semáforos de Linux para garantizar la exclusión mutua al acceder a la cuenta regresiva. No requerirá otro tipo de sincronización más avanzada. Como ejemplo de utilización estudie la solución de *Mem*.

Antes de cargar y probar su tarea asegúrese de ejecutar el comando Linux *sync* para garantizar que sus archivos hayan sido grabados en disco y no están pendientes en un caché de Linux. Recuerde que los errores en su driver pueden hacer que Linux se bloquee indefinidamente y tenga que reiniciar el sistema operativo.

Ayuda

En la función *read*, para convertir la cuenta regresiva *c* a la línea que Ud. debe entregar en *buf*, use:

```
#define KBUF_SIZE 20
char kbuf[KBUF_SIZE];
int len= snprintf(kbuf, KBUF_SIZE, "%d\n", c);
if (count>len)
  count= len;
```

Luego use *copy_to_user* para transferir *count* bytes desde *kbuf* a *buf*, siendo *buf* el parámetro recibido por *read*. Su función *read* debe retornar el último valor de *count*. No olvide sumar los bytes leídos a **f_pos*. Si al inicio de *read*, la cuenta regresiva es 0 o **f_pos* es distinto de 0, retorne 0 bytes leídos, es decir fin de archivo.

En la función *write*, para leer un número entero en base 10 contenido en *count* bytes en la dirección *buf* del área del usuario, use:

```
int c;
int rc= kstrtoint_from_user(buf, count, 10, &c);
```

Si *buf* contiene un número válido, su valor quedará almacenado en la variable *c*, *rc* será 0, y su función *write* debe retornar el mismo valor *count* que recibió como parámetro. Si *buf* no contiene un número válido, *rc* será un código de retorno negativo que indica que el argumento es inválido. Su función *write* debe retornar *rc* en ese caso.

Cuidado: Si *kstrtoint_from_user* retorna un valor negativo, fracasó la escritura y retornan ese mismo valor. Pero si *kstrtoint_from_user* retorna 0, la escritura es exitosa y ¡deben retornar *count*! Si retornan 0, el núcleo se va a quedar en un ciclo escribiendo con write hasta que se escriban los *count* bytes solicidados. Esto nunca ocurrirá. Para colmo, como les recomiendo usar *printk* para reportar las operaciones, eso va a llenar el disco con esos mensajes. Quizás no puedan volver a hacer andar la interfaz gráfica debido al disco lleno.

Notas importantes

- Para cargar el módulo en el núcleo Ud. debe tener acceso a una máquina con Linux en donde pueda convertirse en el usuario *root*. Si no es el caso, pero dispone de un PC con windows o MacOS, siga las instrucciones de esta página para instalar una máquina virtual con Debian/Linux. Para convertirse en *root* en esta máquina use este comando en un terminal de Debian: ***sudo bash***
- Asegúrese de que estén instalados los encabezados del núcleo. En las distribuciones derivadas de Debian, esto se logra con: *sudo apt update ; sudo apt-get install linux-headers-$(uname -r)*
- La ruta al directorio de trabajo no puede contener un nombre que incluya espacios en blanco. Por ejemplo, no podrán compilar en el directorio: */home/fulano/sistemas operativos/*
- El directorio de trabajo debe estar en el sistema de archivos de Debian (no puede ser de Windows).
- Podría ocurrir que la carga del módulo en el núcleo falle con uno de estos errores:

  insmod: ERROR: could not insert module remate.ko: required key not available

  insmod: ERROR: could not insert module remate.ko: Operation not permitted

Se debe a que el PC que están usando tiene activado el modo "*secure boot*" a nivel de la BIOS. Tienen que desactivarlo mientras estén haciendo esta tarea. Para desactivarlo tienen que entrar al menú de la BIOS justo después de encender el computador y antes de que parta el sistema operativo. Esto es altamente dependiente del modelo de PC que usan, de modo que no los puedo ayudar más. Después de hacer la tarea vuelvan a activarlo para mejorar la seguridad del computador.

- No es posible hacer esta tarea bajo ninguna de las versiones de *Windows Subsystem for Linux*.

Entrega

La tarea se entrega *funcionando* en U-cursos. Para ello entregue solo el archivo *cdown.c* que implementa el driver pedido. Se descontará medio punto por día de atraso, excepto días sábado, domingo o festivos.